Recitation 10: NP-Completeness

April 14, 2005

Vinod Vaikuntanathan

Readings: Sections 7.4, 7.5

Outline for Today: Let's look back at what we did this week.

1. \( SAT = \{ (\phi) \mid \phi \text{ is a satisfiable Boolean formula} \} \)

2. Cook-Levin Theorem: \( SAT \in P \) iff \( P=NP \). That is, \( SAT \) is NP-complete. Review proof of Cook-Levin Theorem.

3. What about coNP-completeness? Show that the complement of \( SAT \) is coNP-complete. The \( NP \supseteq coNP \) question is quite relevant in practice too. Consider the problem of program-checking.

Problem 1: Let \( HALF - CLIQUE = \{ (G) \mid G \text{ is an undirected graph having a clique of size at least } n/2, \text{ where } n \text{ is the number of vertices in } G \} \). Show that \( HALF - CLIQUE \) is \( NP \)-complete. (Build on the \( CLIQUE \) problem).

Problem 2: (Sipser 7.29) Show that, if \( P = NP \), a polynomial time algorithm exists that, given a Boolean formula \( \phi \), actually produces a satisfying assignment for \( \phi \), if it is satisfiable. (Note: NP is a class of languages and this problem is the description of a function, that takes a formula \( \phi \) and produces a satisfying assignment if \( \phi \) is satisfiable, and a special symbol \( \bot \) if it is not.)

If we get time, we will do this fun problem too.

Problem 3:

1. Show that \( UNARY-PRIMES=\{1^n \mid n \text{ is a prime number} \} \) is in \( P \). (Hmm, this is cheating!)

2. Show that \( PRIMES=\{n \mid n \text{ is a prime number in binary}\} \) is in \( NP \) and \( coNP \). (And actually Agarwal, Saxena, and Kayal recently showed that it is also in \( P \)!)